WEST Search History

DATE: Friday, June 27, 2003

Set Name side by side	Query	Hit Count	Set Name result set
	DB=USPT; PLUR=NO; OP=ADJ		
L18	6122641.pn.	1	L18
L17	l14 and l12	1	L17
L16	L15 and l12	0	L16
L15	stor$4 near3 l14	8	L15
L14	"object attribute data"	42	L14
L13	object adj1 attribute$1	1274	L13
L12	L11 and l4	38	L12
L11	L10 or l8	1796	L11
L10	((707/100)!.CCLS.)	1220	L10
L9	(707/103$2).ccls	0	L9
L8	((707/103R)!.CCLS.)	715	L8
L7	(707/103R).ccls	0	L7
L6	(707/103$2).ccls	0	L6
L5	((707/103)!.CCLS.)	0	L5
L4	l1 or l2	100	L4
L3	l1 and l2	0	L3
L2	(6151604 or 6163775 or 6175836 or 6182121 or 6208992 or 6236988 or 6341289 or 6442566 or 6484180 or 6490581 or 6199195 or 5499371 or 5504885 or 5615112 or 5717924 or 5884311 or 6067548 or 5737597 or 5809508 or 5226111 or 5481700 or 6321374 or 5335346 or 6240422 or 5261093 or 5752253 or 6366904 or 5197005 or 6240423 or 6405198 or 5809506 or 6105026 or 6128621 or 6370529 or 6272495 or 6457020 or 6128627 or 6128623 or 6209003 or 5721925 or 6336137 or 6393424 or 6430556 or 6163776 or 5551029 or 5826268 or 6006234 or 5819254 or 5870751 or 6349343).pn.	50	L2
L1	(6021410 or 5937409 or 6418448 or 6026408 or 5604892 or 4893232 or 6477527 or 6463433 or 5548749 or 5809297 or 5819086 or 5550971 or 5495604 or 5574908 or 5590322 or 5592668 or 5315709 or 5907847 or 6112210 or 6173290 or 6112207 or 5732257 or 5799310 or 6047291 or 6078925 or 6519603 or 6356946 or 5915115 or 6467085 or 6035303 or 5907844 or 5813002 or 6289358 or 6292880 or 6453319 or 5873093 or 6122641 or 5560005 or 5724575 or 6108664 or 5761493 or 5826077 or 5893087 or 5440730 or 5664177 or 5729730 or 5734887 or 5797137 or 5905982 or 6012067).pn.	50	L1

END OF SEARCH HISTORY

WEST

End of Result Set

Generate Collection | Print

L17: Entry 1 of 1 File: USPT Sep 2, 1997

DOCUMENT-IDENTIFIER: US 5664177 A
TITLE: Data processing system having a data structure with a single, simple primitive

US Patent No. (1):
5664177

Brief Summary Text (14):
The data structure further comprises a single holder attribute data object for each of the attribute data objects. Each of the holder attribute data objects is chosen from the plurality of attribute data objects such that a being-held relationship exists between each attribute data object and its holder attribute data object and such that a hierarchy of the attribute data objects is established by ensuring that each of the attribute data objects has a being-held relationship with only a single other attribute data object.

Detailed Description Text (23):
The hierarchical arrangement of attribute data objects gives rise to a "containment tree," which is used to contain the attribute data objects that collectively represent conceptual objects. A containment tree of an attribute data object, called the root attribute data object or root of the containment tree, includes all attribute data object, held by the root (i.e., directly held), as well as all attribute data objects held by any other attribute data objects in the containment tree (i.e.,, indirectly held). An attribute data object held directly or indirectly by another attribute data object is said to be "contained" by that other attribute data object or in that other attribute data object's containment tree. In a containment tree, the conceptual object represented by that tree corresponds to the root. All other attribute data objects in the containment tree represent conceptual sub-objects, such as components, listed items, or relationships within the conceptual object represented by the tree. The attribute data objects in a containment tree may also represent relationships with attribute data objects outside the containment tree.

Current US Original Classification (1):
707/100

CLAIMS:

1. A memory for storing data for access by an application program being executed on a data processing system, comprising:

a data structure stored in said memory, said data structure including information resident in a database used by said application program and including:

a plurality of attribute data objects stored in said memory, each of said attribute data objects containing different information from said database;

a single holder attribute data object for each of said attribute data objects, each of said holder attribute data objects being one of said plurality of attribute data objects, a being-held relationship existing between each attribute data object and its holder attribute data object, and each of said attribute data objects having a being-held relationship with only a single other attribute data object, thereby establishing a hierarchy of said plurality of attribute data objects;

a referent attribute data object for at least one of said attribute data objects, said referent attribute data object being nonhierarchically related to a holder attribute data object for the same at least one of said attribute data objects and also being one of said plurality of attribute data objects, attribute data objects for which there exist only holder attribute data objects being called element data objects, and attribute data objects for which there also exist referent attribute data objects being called relation data objects; and

an apex data object stored in said memory and having no being-held relationship with any of said attribute data objects, however, at least one of said attribute data objects having a being-held relationship with said apex data object.

6. A data processing system executing an application program and containing a database used by said application program, said data processing system comprising:

cpu means for processing said application program; and

memory means for holding a data structure for access by said application program, said data structure being composed of information resident in said database used by said application program and including

a plurality of attribute data objects stored in said memory, each of said attribute data objects representing different information from said database;

a single holder attribute data object for each of said attribute data objects, each of said holder attribute data objects being one of said plurality of attribute data objects, a being-held relationship existing between each attribute data object and its holder attribute data object, and each of said attribute data objects having a being-held relationship with only a single other attribute data object thereby establishing a hierarchy of said plurality of attribute data objects;

a referent attribute data object for at least one of said attribute data objects, said referent attribute data object being nonhierarchically related to a holder attribute data object of the same at least one of said attribute data objects and also being one of said plurality of attribute data objects, attribute data objects for which there exist only holder attribute data objects being called element data objects, and attribute data objects for which there also exist referent attribute data objects being called relation data objects; and

an apex data object stored in said memory and having no being-held relationship with any of said attribute data objects, however, at least one of said attribute data objects having a being-held relationship with said apex data object.

18. The data processing system of claim 17 wherein said means for removing also includes means for removing from said data structure all attribute data objects which have a being-held relationship with said specified attribute data object and all attribute data objects which have a being-held relationship with any attribute data object removed from said data structure by said removing means.

20. In a data processing system executing an application program, wherein said application program is accessing a data structure composed of information resident in a database used by said application program, wherein said data structure resides in a memory of said data processing system and includes a plurality of attribute data objects each representing different information from said database, wherein for each of said attribute data objects there exists a holder attribute data object and for certain of said attribute data objects there also exists a referent attribute data object related to a holder attribute data object for that attribute data object, each of said holder and referent attribute data objects being one of said plurality of attribute data objects, and wherein each of said plurality of attribute data objects has a being-held relationship with its holder attribute data object thereby establishing a hierarchy of said plurality of attribute data objects, a method of accessing attribute data objects in said data structure comprising the steps of:

selecting, by said application program, information resident in said database;

searching, by a data management program executed by said data processing system, said data structure for one of said attribute data objects representing said selected information;

retrieving, by said data management program, the attribute data objects having a being-held relationship with said one of said attribute data objects representing said selected information; and

transmitting to said application program by said data management program, information related to said retrieved attribute data objects.

21. In a data processing system executing an application program, wherein said application program is accessing a data structure composed of information resident in a database used by said application program, wherein said data structure resides in a memory of said data processing system and includes a plurality of attribute data objects each representing different information from said database, wherein for each of said attribute data objects there exists a holder attribute data object and for certain of said plurality of attribute data objects there also exists a referent attribute data object related to a holder attribute data object for that attribute data object, each of said holder and referent attribute data objects being one of said plurality of attribute data objects, and wherein each of said attribute data objects has a being-held relationship with its holder attribute data object thereby establishing a hierarchy of said plurality of attribute data objects, a method of accessing attribute data objects in said data structure comprising the steps of:

selecting, by said application program, information resident in said database;

searching, by a data management program executed by said data processing system, said data structure for one of said attribute data objects representing said selected information;

retrieving, by said data management program, a referent attribute data object for said attribute data object representing said selected information; and

transmitting to said application program by said data management program, information related to said retrieved referent attribute data object.

22. In a data processing system executing an application program, wherein said application program is accessing a data structure composed of information resident in a database used by said application program, wherein said data structure resides in a memory of said data processing system and includes a plurality of attribute data objects each representing different information from said database, wherein for each of said attribute data objects there exists a holder attribute data object and for certain of said attribute data objects there also exists a referent attribute data object related to a holder attribute data object for that attribute data object, each of said holder and referent attribute data objects being one of said plurality of attribute data objects, and wherein each of said attribute data objects has a being-held relationship with its holder attribute data object thereby establishing a hierarchy of said plurality of attribute data objects, a method of creating attribute data objects in said data structure comprising the steps of:

selecting, by said application program, information to be entered into said database;

creating, by a data management program executed by said data processing system, an attribute data object for said selected information;

choosing, by said data management program, one of said attribute data objects as a holder attribute data object for said created attribute data object, the chosen holder attribute data object being chosen according to said selected information; and

entering, by said data management program, the created data object into said data structure.

WEST

Generate Collection | Print

L15: Entry 1 of 8 File: USPT Mar 25, 2003

DOCUMENT-IDENTIFIER: US 6539388 B1
TITLE: Object-oriented data storage and retrieval system using index table

Detailed Description Text (152):
The index management section 410 manages a correspondence among a given index, object, attribute data, and data storage position and performs processing necessary for access while referring to necessary tables in accordance with the type of data access using various indices. In the following description, [ ] represents a class name, and { } represents an instance name.

Detailed Description Text (337):
(6) A data retrieval method using a data storage and retrieval system for registering an object corresponding to stored data and managing data of each data component obtained by dividing internal data of the stored data into arbitrary storage sections using attributes of the object, comprising: an index retrieval step of retrieving a data component in accordance with a given index describing a path for access to the data component using object attribute data and data storage position data; an object retrieval step of outputting corresponding attribute data on the basis of the given description, using class data containing one or a plurality of attribute data of the object and:instance data belonging to the class; and an data access step of accessing the data component in the stored data on the basis of the data storage position data about the data component, wherein the index retrieval step comprises a metaindex retrieval step of expanding the given index description to an index of primary level using a correspondence between a metaindex secondarily added to the index and the index as a base of the metaindex, and the data component is accessed on the basis of the given metaindex by using the attribute of the:object represented by the correspondence.

Detailed Description Text (341):
(10) A data retrieval method using a data storage and retrieval system for registering an object corresponding to stored data and managing data of each data component obtained by dividing internal data of the stored data into arbitrary storage sections using attributes of the object, comprising: an index retrieval step of retrieving a data component in accordance with a given index describing a path for access to the data component using a storage position management table in which a correspondence between object attribute data and storage position designation of the data component is recorded; a metaindex retrieval step of expanding a given index description to an index of primary level as needed using a metaindex table in which a correspondence between a metaindex secondarily added to the index and the index as a base of the metaindex is recorded; an indirect data index retrieval step of accessing the data component using an object attribute represented by an indirect data index using the data component which is indirectly pointed from another attribute in one object or an attribute of another object through the relation, and calculating contents of a virtual data component from the obtained data component in accordance with a description of the indirect data index on an attribute definition table in which one or a plurality of attribute data of the object are recorded; an overwrite step of calculating an attribute value to be written when the given object attribute is writable; an object retrieval step of outputting corresponding attribute data on the basis of the given description, using at least the attribute definition table, from class data of the object and instance data belonging to the class; a data access step of accessing the data component in the stored data on the basis of the data storage position data about the data component; a change detection step of monitoring a change in the data component and detecting the change; a change notification step of

notifying a change reflection destination object of contents of the change upon receiving notification of detection of the change; a change reflection step of changing an attribute value of the change reflection destination object in accordance with contents of notification; an interclass data translation step of generating a new instance corresponding to a new class on the basis of a translation rule in response to generation of an instance in an existing class using an interclass data translation rule table in which the object translation rule describing a relation between the new class and the existing class is recorded; a plug-in step of dynamically performing translation between an object attribute represented by the index given from an external application and an object attribute registered for an existing object on the basis of metaindex data and/or interclass translation rule data; an input/output step of externally accessing the contents of the object; and a display step of displaying the contents of the object.

L15: Entry 6 of 8 File: USPT Sep 15, 1998

DOCUMENT-IDENTIFIER: US 5809507 A
TITLE: Method and apparatus for storing persistent objects on a distributed object network using a marshaling framework

Abstract Text (1):
Data structures, methods and devices for implementing persistence data storage such that persistent objects may be efficiently created and accessed in a distributed client/server computing system are disclosed. In one aspect of the invention, a method for managing persistence data for installed persistent objects involves marshaling a persistent object attribute value into a marshal buffer to provide an encoded persistent object attribute value, updating the persistent object attribute value to provide an updated persistent object attribute value, unmarshaling the updated persistent object attribute value from the marshal buffer to provide a decoded updated persistent object attribute value, and writing the decoded updated persistent object attribute value to the data store. In another aspect of the invention, a method for writing decoded updated persistent object attribute values to the data store includes extracting an index of persistent object attributes stored in the data structure, finding the location of the persistent object attribute in the data base, adding the persistent object attribute to the data store if the persistent object attribute cannot be found in the index, and writing the persistent object attribute value to the data store at the location corresponding to the persistent object attribute.

Brief Summary Text (18):
In one aspect of the invention, a method for managing persistence data for installed persistent objects involves marshaling a persistent object attribute value into a marshal buffer to provide an encoded persistent object attribute value, updating the persistent object attribute value to provide an updated persistent object attribute value, unmarshaling the updated persistent object attribute value from the marshal buffer to provide a decoded updated persistent object attribute value, and writing the decoded updated persistent object attribute value to the data store. In one embodiment, the method includes retrieving the persistent object attribute value from the persistent object prior to the step of marshaling the persistent object attribute value into the marshal buffer. In another embodiment, the method includes opening the data store, extracting from the data store an index of persistent object attributes contained in the data store, searching the index of persistent object attributes contained in the data store to determine the location of the persistent object attribute in the data store, and returning the persistent object attribute value corresponding to the persistent object attribute.

Brief Summary Text (19):
In another aspect of the invention, a method for writing decoded updated persistent object attribute values to the data store includes extracting an index of persistent object attributes stored in the data structure, finding the location of the persistent object attribute in the data base, adding the persistent object attribute to the data store if the persistent object attribute cannot be found in the index, and writing the persistent object attribute value to the data store at the location corresponding to the persistent object attribute.

Brief Summary Text (20):
In still another aspect of the invention, a method for managing persistence data for the installed persistent objects on a distributed object system includes invoking a method on the distributed object system, the method being associated with a persistent object attribute having a persistent object attribute value and a persistent object

attribute name, marshaling the persistent object attribute value into a marshal buffer to provide thereby an encoded persistent object attribute value, updating the persistent object attribute value to provide an updated persistent object attribute value, unmarshaling the updated persistent object attribute value from the marshal buffer to provide a decoded updated persistent object attribute value, and writing the decoded updated persistent object attribute value to the data store. In one embodiment, the method includes retrieving the persistent object attribute value from the persistent object prior to the step of marshaling the persistent object attribute value into the marshal buffer. In another embodiment, writing the decoded updated persistent object attribute value to the data store includes extracting an index of persistent object attributes stored in the data store, finding the location of the persistent object attribute in the data base using the persistent object attribute name and the index, adding the persistent object attribute to the data store if the persistent object attribute cannot be found in the index, writing the persistent object attribute name to the data store at the location corresponding to the persistent object attribute, and writing the persistent object attribute value to the data store at the location corresponding to the persistent object attribute.

CLAIMS:

3. The method of claim 1, further including the step of retrieving the persistent object attribute value corresponding to the persistent object attribute from the data store prior to the step of marshaling the persistent object attribute value into the marshal buffer.

4. The method of claim 3, wherein said step of retrieving the persistent object attribute value corresponding to the persistent object attribute from the data store includes the steps of:

a) opening the data store;

b) extracting an index of persistent object attributes contained in the data store from the data store;

c) searching the index of persistent object attributes contained in the data store to determine the location of the persistent object attribute in the data store;

d) identifying the location of the persistent object attribute in the data store; and

e) returning the persistent object attribute value corresponding to the persistent object attribute.

7. The method of claim 6, further including the step of unmarshaling persistence data associated with the persistent object attribute from the data store file.

8. The method of claim 1, wherein the step of writing the decoded updated persistent object attribute value to the data store comprises the steps of:

a) extracting an index of persistent object attributes stored in the data store;

b) determining whether the persistent object attribute is located in the data store using the index of persistent object attributes;

c) adding the persistent object attribute to the data store if it is determined that the persistent object attribute is not located in the index of persistent object attributes; and

d) writing the persistent object attribute value to the data store at location corresponding to the persistent object attribute when it is determined that the persistent object attribute is located in the data store.

12. The method of claim 11, wherein the step of retrieving the persistent object attribute value from the data store includes the steps of:

a) opening the data store;

b) extracting from the data store an index of persistent object attributes, the index of persistent object attributes being contained in the data store;

c) searching the index of persistent object attributes contained in the data store using the persistent object attribute name to determine a location of the persistent object attribute in the data store; and

d) returning the persistent object attribute value corresponding to the persistent object attribute.

13. The method of claim 12 wherein the step of opening the data store includes the steps of:

a) selecting an appropriate data store file;

b) creating the marshal buffer using the appropriate data store file;

c) unmarshaling a data store version record corresponding to the data store file;

d) determining whether the data store file is supported as the data store;

e) creating a data store root object associated with the data store when it is determined that the data store file is supported as the data store;

f) unmarshaling the persistence object attribute value associated with the persistent object attribute from the data store file; and

g) registering the data store with a data store registry.

14. The method of claim 9, wherein the step of writing the decoded updated persistent object attribute value to the data store includes the steps of:

a) extracting an index of persistent object attributes stored in the data store;

b) finding the location of the persistent object attribute in the data store using the persistent object attribute name and the index;

c) adding the persistent object attribute to the data store when the persistent object attribute is not found in the index;

d) and writing the persistent object attribute value to the data store at the location corresponding to the persistent object attribute when the persistent object attribute is found in the index.

18. The distributed object system arranged to store persistence data for installed persistent objects in a data store as recited in claim 15 further including:

a locator for finding the location of the persistent object attribute in the data store; and

an appending mechanism for adding the persistent object attribute to the data store when the persistent object attribute is not found in the data store.